An Ant Colony Optimization and Hybrid Metaheuristics Algorithm to solve the Split Delivery Vehicle Routing Problem

Authors: Gautham Rajappa, Joseph Wilck, John Bell

Abstract
Split Delivery Vehicle Routing Problem (SDVRP) is a relaxation of the Capacitated Vehicle Routing Problem (CVRP) wherein a customer can be visited by more than one vehicle. Two approaches using, 1) Ant Colony Optimization and 2) hybrid metaheuristics algorithm comprising a combination of ACO, Genetic Algorithm (GA) and heuristics are proposed and tested on benchmark SDVRP problems. The results indicate that the two proposed algorithms are competitive in both solution quality and solution time. In some instances, the best ever solutions have been found for particular problem instances.

Keywords
Optimization, Metaheuristics, Ant Colony Optimization, Split Delivery Vehicle Routing Problem, Genetic Algorithm

1. Introduction
The Vehicle Routing Problem (VRP) is a prominent problem in the areas of logistics, operations research, and transportation management. With an objective to minimize the delivery cost of goods to a set of customers from depot(s), numerous variants of the VRP have been developed and studied over the years. One such variant is the Capacitated Vehicle Routing Problem (CVRP). The objective of a CVRP is to minimize cost of delivering a single product to a set of customers from a single depot using a homogenous fleet of vehicles (Liu et al., 2009). The Split Delivery Vehicle Routing Problem (SDVRP) is a relaxation of the Capacitated Vehicle Routing Problem (CVRP). In the case of a CVRP, each customer is served by only one vehicle, whereas in SDVRP, the customer demand can be split between vehicles. For example, consider three customers each with a demand of 100 served by vehicle with a capacity of 150. In the case of

the CVRP, three vehicles are required but in the case of SDVRP, since the customer demand can be split amongst multiple vehicles, only two vehicles are required to fulfill the customer demand.

SDVRP was first developed by Dror and Trudeau (1989; 1990). They showed that if the demand is relatively low compared to the vehicle capacity and the triangular inequality holds, an optimal solution exists in the SDVRP in which two routes cannot have more than one common customer. In addition, it was proven that the SDVRP is a NP-hard problem, and that there are potential savings in solving instances of the problem in terms of both minimizing the total distance traveled in serving all demands, as well as the number of vehicles used.

Over the past few years, several heuristic methods have been applied to solve the SDVRP, such as a construction heuristic (Wilck and Cavalier, 2012a), a genetic algorithm (Wilck and Cavalier, 2012b), and Tabu search (Archetti et al., 2006). An IEEE conference proceeding paper by Sui et al. (2008) presents an ACO approach for the SDVRP, but does not present empirical results compared to published methods. Hence, we developed an ACO approach for the SDVRP and further, extended the ACO algorithm to develop a hybrid metaheuristics algorithm in which the initial set of population (vehicle routes) is generated using ACO. Then, a combination of heuristics and genetic algorithm is applied to discover a more optimal vehicle route. We tested the capability of the two proposed algorithms on different benchmark test problems in order to measure the ability of these algorithms to generate competitive solutions for this problem.

The rest of the paper is organized as follows: Section 2 and Section 3 provide an overview of the SDVRP and our proposed algorithms respectively. Computational experiments are described in Section 4, and the conclusions and future research opportunities are summarized in Section 5.

2. SDVRP Formulation, Literature Review, and Benchmark Data Sets

This section of the paper is organized as follows: subsection 1 presents the SDVRP formulation, subsection 2 and 3 presents a literature review for the SDVRP and application of metaheuristics in solving the VRP respectively, and subsection 4 provides an overview of the benchmark data sets for the SDVRP.

2.1 SDVRP Formulation

According to Aleman et al. (2010), the SDVRP is defined on an undirected graph $G = (V, E)$ where V is the set of n nodes of the graph and $E = \{(i, j) : i, j \in V, i < j\}$ is the set of edges connecting the nodes . Node 1 represents a depot where a fleet M of identical vehicles with capacity Q are stationed, while the remaining node set $N = \{2, \ldots, n\}$ represents the customers. A non-negative cost, usually a function of distance or travel time, c_{ij} is associated with every edge (i, j). Each customer $i \in N$ has a demand of q_i units. The optimization problem is to determine which customers are served by each vehicle and what route the vehicle will follow to serve those assigned customers, while minimizing the operational costs of the fleet, such as travel distance, gas consumption, and/or vehicle depreciation. The most frequently used formulations for SDVRP found in literature are from Dror and Treadeau (1990), Frizzell and Giffin (1992), and Dror et al. (1994).

In this research, we use the SDVRP flow formulation adapted from Wilck and Rajappa (2010), which is given below. This formulation assumes that c_{ij} satisfies the triangle inequality and that exactly the minimum number of vehicle routes, K , are used. The formulation does not assume that distances are symmetric.

Indexed Sets:

$i = \{1, 2, \ldots, n\}$; node index ; 1 is the depot

$j = \{1, 2, \ldots, n\}$; node index

$k = \{1, 2, \ldots, m\}$; route index

Parameters:

m : The number of vehicle routes

n : The number of nodes

Q : The vehicle capacity

c_{ij} : The cost or distance from node i to node j

d_i : The demand of customer i , where $d_1 = 0$.

Decision Variables:

x_{ijk}: A binary variable that is one when arc (i, j) is traversed on route k; zero otherwise

u_{ik}: Free variable used in the sub-tour elimination constraints

y_{ik}: A binary variable that is one when node i is visited on route k; zero otherwise

v_{ik}: A variable that denotes the amount of material delivered to node i on route k

Without loss of generality, y_{ik} and v_{ik} are not defined for $i = 1$.

Objective: Minimize Travel Distance

Minimize $$Z = \sum_{i=1}^{n} \sum_{\substack{j=1 \\ i \neq j}}^{n} c_{ij} \sum_{k=1}^{m} x_{ijk} \quad \textbf{(1)}$$

Constraints:

$$\sum_{k=1}^{m} v_{ik} = d_i, \forall\, i = 2, ..., n \quad \textbf{(2)}$$

$$\sum_{i=2}^{n} v_{ik} \leq Q, \forall\, k = 1, ..., m \quad \textbf{(3)}$$

$$\sum_{\substack{i=1 \\ i \neq p}}^{n} x_{ipk} - \sum_{\substack{j=1 \\ j \neq p}}^{n} x_{pjk} = 0, \forall\, k = 1, \ldots, m;\ p = 1, \ldots, n \quad \textbf{(4)}$$

$$u_{ik} - u_{jk} + n x_{ijk} \leq n - 1,\ \forall\, i = 2, ..., n;\ i \neq j;\ k = 1, ..., m \quad \textbf{(5)}$$

$$d_i y_{ik} \geq v_{ik}, \forall k = 1, ..., m; i = 2, ..., n \quad \textbf{(6)}$$

$$\sum_{\substack{j=1 \\ j \neq i}}^{n} x_{ijk} = y_{ik}, \forall\, k = 1, ..., m;\ i = 2, ..., n \quad \textbf{(7)}$$

$$\sum_{j=2}^{n} \left(x_{1jk} + x_{j1k} \right) = 2, \forall\, k = 1, ..., m \quad \textbf{(8)}$$

$$x_{ijk} \in \{0,1\}, \forall i = 1, \ldots, n;\ j = 1, \ldots, n;\ i \neq j;\ k = 1, \ldots, m \quad \textbf{(9)}$$

$$y_{ik} \in \{0,1\}, \forall\, i = 2, \ldots, n;\ k = 1, \ldots, m \quad \textbf{(10)}$$

$$v_{ik} \geq 0, \forall i = 2, \ldots, n;\ k = 1, \ldots, m \quad \textbf{(11)}$$

The objective is represented by Equation (1), which minimizes the total distance traveled. Constraints (2) and (3) ensure that all customer demand is satisfied without violating vehicle capacity. Constraints (4) and (5) ensure flow conservation and that sub-tours are eliminated, respectively. The sub-tours are eliminated using the method described in Miller et al. (1960). Constraints (6) and (7) force the binary variables to be positive if material is delivered to node i on route k. Constraint (8) ensures that the depot is entered and exited on every vehicle route, and constraints (9) – (11) provide variable restrictions.

2.2 SDVRP Literature Review

In recent work on the SDVRP, several researchers developed approaches for generating solutions to the SDVRP. Archetti et al. (2006) developed a Tabu search algorithm called SPLITTABU to solve the SDVRP in which they showed that there always exists an optimal solution where the quantity delivered by each vehicle when visiting a customer is an integer number. Also, Archetti et al. (2008a) performed a mathematical analysis and proved that by adopting a SDVRP strategy, a maximum of 50% reduction can be achieved in the number of routes. They also showed that when the demand variance is relatively small and the customer demand is in the range of 50% to 70% of the vehicle capacity, maximum benefits are achieved by splitting the customer's demand. Furthermore, Archetti et al. (2008b) presented a solution approach that combines heuristic search and integer programming. Boudia et al. (2007) solved an SDVRP instance using a memetic algorithm with population management which produced better and faster results than the SPLITTABU approach (Archetti et al., 2006). Mota et al. (2007) proposed an algorithm based on scatter search methodology which generated excellent results compared to SPLITTABU. Archetti and Sperenza (2012) have published an extensive survey on SDVRP and its variants.

2.3 Metaheuristics for the VRP: Literature Review

Over a period of time, researchers have developed numerous metaheuristics based solutions for VRP and its variants. One of the first papers on application of ACO in VRP was proposed by Bullheimer et al. (1997; 1999). They proposed a variant called "hybrid ACO" using 2-opt heuristic. Their algorithm was tested on fourteen Christofides benchmark problems and computation results showed that the results obtained were not as good as the ones obtained from other metaheuristics. Additionally, Gambardella et al. (1999) proposed an algorithm based on

ACO called MACS-VRPTW (Multiple Ant Colony System for Vehicle Routing Problems with Time Windows). This is the first paper in which a multi-objective minimization problem is solved using a multiple ant colony optimization algorithm. MACS-VRPTW not only provided improved solutions on benchmark test problems but also was on par or better than other existing methods in terms of solution quality and computation time. Next, Baran and Schaerer (2003) proposed a multi objective ACO for VRPTW based on MACS-VRPTW but instead of using two ant colonies, only one ant colony was used to find a set of Pareto optimal solutions for three objectives. Yu and Yang (2011) proposed an improved ACO (IACO) to solve period vehicle routing problem with time windows (PVRPTW) in which the planning horizon is extended to multiple days and deliveries are made within a specific time window for each customer. A combination of multi-dimension pheromone information and two crossover operations (one-point and two-point crossover) was used to solve PVRPTW and the algorithm was tested on benchmark problems.

Rizzoli et al. (2004) have done extensive surveys on ACO for VRP and its variants. Montemanni et al. (2004) proposed an ACO solution called ACS-DVRP to solve the Dynamic VRP (DVRP) in which the large DVRP problem was divided into smaller static VRP problems. Bell et al. (2004) proposed single and multiple ant colony methodologies to solve the VRP. Their experimental results showed that the best results were obtained when the candidate list size was between ten and twenty. Doerner et al. (2004) proposed a parallel ant system algorithm for CVRP and this is the first paper which shows the effect of parallelization of processors on speed and efficiency. Additionally, Favaretto et al. (2007) formulated and provided an ACO based solution for VRP with multiple time windows and multiple visits which consider periodic constraints. Computation results show that their proposed algorithm provides better solutions as compared to some of the other metaheuristics published in the literature.

In the area of transportation management, Yi and Kumar (2007) proposed an ACO based approach for solving a logistics problem that involves supplying goods to distribution centers and evacuating injured people to medical centers in disaster relief operations. The ACO algorithm decomposes the emergency logistics problem into a vehicle routing problem and a multi-commodity dispatch problem. The ACO algorithm also includes several trial updating

strategies and was tested on randomly generated test problems. The results of ACO algorithm were compared with solutions from CPLEX which was used to solve the mathematical model of the disaster relief problem. The ACO algorithm generated comparable solutions within one minute of computational time, when compared to the optimal solution generated by CPLEX. The quality and time to generate the ACO generated solutions were considered acceptable due to the real-time needs of during a disaster relief event.

Gajpal and Abad (2009) proposed an ant colony system for VRP with simultaneous delivery and pickup (VRPSDP). Computational results on benchmark test problems show that the proposed algorithm provides better results both in terms of solution quality and CPU time when compared to previously published methods and data sets, including 31 new best known solutions for the VRPSDP data sets. Li, et al. (2009) provided an ant colony optimization metaheuristic that was hybridized with tabu search to find good solutions to the open vehicle routing problem (OVRP). Yu et al. (2011) utilized a parallel ant colony optimization scheme for the virtual multi-depot vehicle routing problem (V-MDVRP). Finally, Hu et al. (2011) provided an ACO based solution for distributed planning problems for home delivery in which a revised methodology to update the pheromone and the probability matrix is proposed.

2.4 SDVRP Benchmark Data Sets

Despite several exact optimization and metaheuristic solution methods being applied to the SDVRP, no previous research has applied either ACO or a hybrid combination of ACO, GA and a heuristic algorithm to solve the SDVRP. Hence, we compare our algorithms in this research with two previous approaches used on the SDVRP that have established benchmark problem instances.

First, Jin et al. (2008) proposed a column generation approach to solve SDVRP with large demands, in which the columns have route and delivery amount information and a limited-search-with-bound algorithm is used to find the lower and upper bounds of the problem. They used column generation to find lower bounds and an iterative approach to find upper bounds for a SDVRP. They also suggested that their approach of solving the SDVRP does not yield good solutions for large customer demands and in such cases, they recommend solving the SDVRP

instance as a CVRP. The number of customers for the 11 test problems from Jin et al. (2008) ranged from 50 to 100, with an additional node for the depot. The data sets also differ by amount of spare capacity (i.e., additional vehicle capacity, accumulated from all vehicles, after serving all customer demand, across all customers). The customers were placed randomly around a central depot and demand was generated randomly based on a high and low threshold.

Second, Derigs et al. (2010) extended four different moves: 2-OPT, EXCHANGE and RELOCATE and RELOCATE 1 of VRP to the SDVRP. They then embedded these four moves in metaheuristics such as simulated annealing (SA), threshold accepting (TA), record-to-record travel (RRT), attribute-based hill climber (ABHC) and attribute based local beam search (ABLBS) and tested their algorithms on datasets from Archetti et al. (2008b) and Chen et al. (2007). Derigs et al. (2010) concluded that the best solutions were obtained using attribute-based hill climber (ABHC).

In Section 4, we compare the results of our two proposed algorithms first with the results of Jin (2008) for their 11 benchmark problems, and then with the solutions from Derigs et al. (2010), which were accomplished on the original 21 problems from Chen et al. (2007). The nodes in the 21 test problems ranged from eight to 288, with an additional node for the depot. The problems do not have any spare vehicle capacity (i.e., additional vehicle capacity, accumulated from all vehicles, after serving all customer demand, across all customers), and the customers were placed on rings (i.e., circular pattern) surrounding a central depot and the demand was either 60 or 90, with a vehicle capacity of 100.

3. Proposed Algorithms

In this section, we describe our proposed algorithms for solving the SDVRP.

3.1 ACO for the SDVRP

Ant Colony Optimization (ACO) is a metaheuristic proposed by Dorigo (1992). Inspired by foraging behavior of ants, ACO belongs to a class of metaheuristic algorithms that can be used to obtain near optimal solutions in reasonable computational time for combinatorial optimization problems. Ants communicate with one another by depositing pheromones, a trace chemical substance that can be detected by other ants (Rizzoli et al., 2004). As ants travel, they deposit

pheromones along their trail, and other ants tend to follow these pheromone trails. However during their journey, ants may randomly discover a new trail, which might be shorter or longer than the previous trail. Pheromones have a tendency to evaporate. Hence, over a period of time, the shortest trail (path) from the food source to the colony will have a larger amount of pheromone deposited as compared with other trails and will become the preferred trail.

The main elements in an ACO are ants that independently build solutions to the problem. For an ant *k*, the probability of it visiting a node *j* after visiting node *i* depend on the two attributes namely:

- **Attractiveness ($\boldsymbol{\eta_{ij}}$):** It is a static component that never changes. In the case of VRP, it is calculated as inverse of arc length for shortest path problems and for other variants, it can depend on other parameters besides the arc length (e.g., in VRPTW it also depends on the current time and the time window limits of the customers to be visited (Rizzoli et al., 2004).
- **Pheromone trails($\boldsymbol{\tau_{ij}}$):** It is the dynamic component which changes with time. It is used to measure the desirability of insertion of an arc in the solution. In other words, if an ant finds a strong pheromone trail leading to a particular node, that direction will be more desirable than other directions. The trail desirability depends on the amount of pheromone deposited on a particular arc (Rizzoli et al., 2004).

For solving a VRP, each individual ant simulates a vehicle. Starting from the depot, each ant constructs a route by selecting one customer at a time until all customers have been visited. Using the formula from Dorigo and Gamberdella (1997), the ant selects the next customer *j* as shown in equation (12):

$$j=\begin{cases} arg\,max\,\{(\tau_{iu})(\eta_{iu}^{\beta})\} & \text{for } u\notin M_k\,, q\leq q_o \\ Equation\ (13), & \text{otherwise} \end{cases} \qquad \textbf{(12)}$$

where τ_{iu} is the amount of pheromone on *arc (i,u), u* being all possible unvisited customers. In classic VRP, locations already visited are stored in ants' working memory M_k and are not considered for selection. However, in the case of SDVRP, the locations for which the demands have not been fulfilled (*demand >0*) are stored in the ants' working memory and are considered

for selection. β establishes correlation between the importance of distance with respect to the pheromone quantity ($\beta > 0$). q is a randomly generated variable between 0 and 1 and q_0 is a predefined static parameter. If equation (12) does not hold, the next customer to be visited is selected based on a random probability rule as shown in equation (13):

$$P_{ij}=\begin{cases} \frac{[(\tau_{ij})]\,[(\eta_{ij}{}^{\beta})]}{\sum_{j\notin M_k}[(\tau_{ij})]\,[(\eta_{ij}{}^{\beta})]} & \text{if } j\notin M_k\text{ , } q>q_o \\ 0\ (depot), & \text{otherwise} \end{cases} \quad \textbf{(13)}$$

If the vehicle capacity constraint is satisfied, the ant will return to the depot before starting the next tour in its route. This selection process continues until all customers are visited by an ant. In ACO, the pheromone trail is updated locally during solution construction and globally at the end of construction phase. An interesting aspect of pheromone trail updating is that every time an arc is visited, its value is diminished which favors the exploration of other non-visited nodes and diversity in the solution. Pheromone trials are updated by reducing the amount of pheromone deposited on each *arc (i,j)* visited by an ant (local update). Also, after a predetermined number of ants construct feasible routes, pheromones are added to all the arcs of the best found solution (global update).

Local update on a particular *arc* (τ_{ij}) is updated done using equation (14) (Dorigo and Gamberdella, 1997) :

$$\tau_{ij} = (1-\alpha)\tau_{ij} + \alpha\tau_0 \quad \textbf{(14)}$$

where $0 \le \alpha \le 1$ is the pheromone trail evaporation rate and τ_0 is the initial pheromone value for all arcs.

Global trial updating is done using equation (15) (Dorigo and Gamberdella, 1997):

$$\tau_{ij} = (1-\alpha)\tau_{ij} + \alpha L^{-1} \quad \textbf{(15)}$$

where L is the best found objective function value (total distance).This procedure is repeated until a terminating condition is met.

3.2 Hybrid Metaheuristics Algorithm

For the hybrid metaheuristics algorithm, the initial set of population (vehicle routes) is generated using ACO and then, a combination of heuristics and genetic algorithm (GA) is applied to discover a more optimal vehicle route.

Genetic algorithms (GA) are population based search algorithms to solve combinatorial optimization problems. It was first proposed by John Holland (Goldberg, 1989). In these algorithms, the search space (population) of a problem is represented as a collection of individuals (chromosomes).Genetic algorithms generate solutions for optimization problem based on theory of evolution using concepts such as reproduction, crossover and mutation. The fundamental concept of a genetic algorithm states a set of conditions to achieve global optima. These conditions describe the reproduction process and ensure that better solution remain in future generations and weaker solutions be eliminated from future generations. This is similar to the Darwin's survival of fittest concept in the theory of evolution. The genetic algorithm search mechanism consists of three phases: (1) Evaluation of fitness function of each solution in the population (2) selection of parent solutions based on fitness values and (3) application of genetic operations such as crossover and mutation to generate new offspring.

A typical genetic algorithm consists of the following steps (Goldberg, 1989):

- ***Step 1:*** *Generate an initial population of N solutions.*
- ***Step 2:*** *Evaluate each solution of the initial population using a fitness function/objective function.*
- ***Step 3:*** *Select solutions as parents for the new generation based on probability or randomness. The best solutions (in terms of fitness or objective) have a higher probability of being selected than poor solutions.*
- ***Step 4:*** *Use the parent solutions from Step 3 to produce the next generation (called offspring). This process is called as crossover. The offspring are placed in the initial set of solutions replacing the weaker solutions.*

- ***Step 5:*** *Randomly alter the new generation by mutation. Usually this is done using a mutation probability.*
- ***Step 6:*** *Repeat Steps 2 through 5 until a stopping criteria is met.*

Due to the constraints of a SDVRP, it is not possible to directly use genetic algorithm in the way it is described above. In particular, after crossover and mutation, there may be solutions which do not satisfy the constraints. Hence, to obtain a feasible set of offspring, we may need to modify the way crossover is done or another possibility is to remove infeasible solutions after mutation and replace them with the solutions having higher fitness value in the old population (Cordeau et al., 2002).

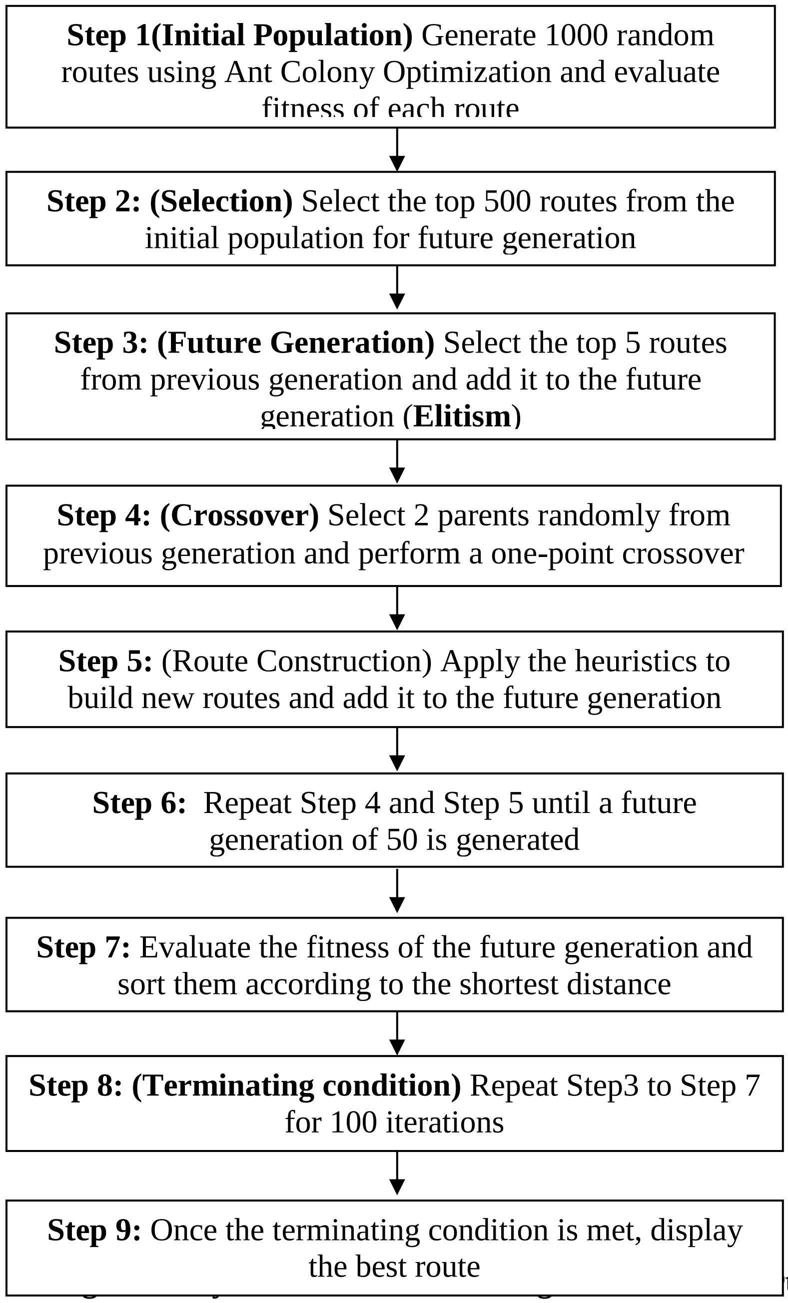
The proposed hybrid algorithm is a combination of Ant Colony Optimization (ACO), Genetic Algorithms and Heuristics, a detailed description of which is given below:

- **Solution encoding:** It's represents a feasible vehicle route. The solutions are encoded as a series of random numbers from 0 to N, wherein, each N represents a node (customer location) and 0 represents a depot. For example, a route is represented as [0,1,2,3,0,3,4,5,0].
- **Initial population:** 1000 random solutions using ant colony optimization metaheuristics are used for the initial population.
- **Fitness:** The objective function (minimizing the total distance) is evaluated for each route from the initial population and then a corresponding fitness value is assigned. The fitness value is the total distance of a particular route.
- **Selection:** Using the fitness value of each route, the top 500 routes from the initial population are selected for future generation.
- **Future Generation (Crossover and mutation):**
    - The size of the future generation is set to 50.
    - Due to the constraints of SDVRP, mutation was not considered.
    - **Elitism:** The top 5 results from previous generation were by default used in the next generation.
    - **Crossover:** Crossover is performed until 50 new routes are generated. Two parents are randomly selected from the previous generation. As described below,

a heuristics based one point crossover is then applied to each of these parents to create future generation.

- **Heuristics:** The routes are constructed as follows:
    - **Condition 1:** For all the available nodes (demand is not satisfied), add the next node to the route if:
        - The node's demand is less than the remaining capacity of the vehicle **and**
        - The next node is closest to the previous node **and**
        - The next node has the largest demand amongst all the nodes.
    - **Condition 2:** If condition 1 is not satisfied, then for all the available nodes (demand is not satisfied), add the next node to the route if:
        - The node's demand is less than the remaining capacity of the vehicle **and**
        - The next node is closest to the previous node.
    - If condition 1 and condition 2 are not satisfied, go back to the depot.
- **Termination condition:** Repeat the Fitness to Heuristics procedure for 100 iterations.

The flowchart for the hybrid metaheuristics algorithm is shown in Figure 1 below:



An IEEE conference proceeding paper by Sui et al. (2008) presents an ACO approach for the SDVRP, but does not present empirical results compared to published methods. To the best of our knowledge and despite previous success in applying ACO and Genetic Algorithms to variants of the VRP, no known research effort has applied just ACO or a combination of ACO and GA to the SDVRP and experimentally tested the ability of the algorithms on SDVRP instances.

4. Computational Experiments

Both algorithms for this study was coded in Java and run on a Windows7, Intel i5 2.4 Ghz, 4 GB RAM computer. For all our test datasets, search parameters were tuned during pilot-testing and set as shown in Table 1 and Table 5. The algorithm was tested against two procedures from the literature, namely Jin et al. (2008) and Derigs et al. (2010) as mentioned in the section on benchmark datasets (Section 2) and each problem was run in 10 separate iterations (Fuellerer et al., 2010). The results are shown in Table 2 and Table 3 for ACO and Table 6 and Table 7 for hybrid metaheuristics algorithm.

4.1 ACO Algorithm

One of the route improvement strategies common to the VRP is to have a candidate list to determine the next location for each customer. In other words, only a set of predetermined closest locations are included in the candidate list for the next possible move. In previous research (Bullnheimer et al., 1999a), set the size of their candidate list to one fourth of the total number of customers, irrespective of the problem size. In pilot testing for this study, we experimented with different candidate list sizes and for our research the candidate list size of one ninth ($n/9$, where n is the number of customers) was found to yield the best solutions for instances of the SDVRP.

Additionally, in the case of CVRP, an ant (vehicle) travels to a customer (node) only if the customer's demand can be completely fulfilled with the remaining vehicle capacity. But in the case of SDVRP a modification to the VRP must be made. Since a customer's demand can be split amongst multiple vehicle routes, an ant can travel to a customer in the ACO SDVRP algorithm based on three conditions: (1) If the customer is in the candidate list, (2) if the customer's demand is not completely fulfilled, and (3) there is remaining capacity on the vehicle. If the above conditions cannot be satisfied for any location, the ant (vehicle) returns to the depot.

Table 1: ACO Parameters

Parameter	Values
α	0.5
β	1.3
τ_0	10^{-5}
q_0	0.9
m (global update counter)	10
Number of iterations	100,000

Table 2: Comparing ACO results versus Jin et al. (2008)

	Ant Colony Optimization				Results from Jin et al.		
Dataset	**Objective Function Mean (std dev))**	**Objective Function (Best)**	**Best Time(s)**	**Total Time(s)**	**Objective Function**	**Total Time(s)**	**GAP**
s51d2	744.03(14.07)	727.28	186.59	699.56	**722.93**	10741	0.60%
s51d3	1001.97(15.87)	982.66	164.5	843.23	**968.85**	833	1.43%
s51d4	1654.56(12.68)	1629.09	1053.95	1074.66	**1605.64**	789	1.46%
s51d5	1416.60(20.37)	1389.01	519.44	1015.48	**1361.24**	10	2.04%
s51d6	2302.72(14.16)	2267.97	584.65	1339.2	**2196.35**	478	3.26%
s76d2	1161.19(12.47)	**1134.27**	**1431.9**	1742.09	1146.68	75074	**-1.08%**
s76d3	1527.25(19.06)	1502.36	979	2078.88	**1474.89**	3546	1.86%
s76d4	2218.51(21.63)	2191.83	337.7	1310.3	**2157.87**	369	1.57%
s101d2	1484.12(16.99)	**1457.39**	**930.81**	3352.49	1460.54	189392	**-0.22%**
s101d3	2000.94(33.52)	**1948.09**	**3166.21**	3938.37	1956.91	36777	**-0.45%**
s101d5	2972.54(17.29)	2945.41	3778.25	4947.82	**2885**	5043	2.09%

*****The objective function values highlighted in bold are the best results

Note: GAP indicates ACO versus best known solution. A negative GAP indicates a new best solution when compared to previous literature.

Jin et al. (2008) Computer Specifications: Pentium 4 Processor, 2.8 GHz CPU, 2 GB memory.

ACO Computer Specifications: Java , Windows7, Intel i5 2.4 Ghz, 4 GB RAM computer.

Table 3: Comparing ACO results versus Derigs et al. (2010)

	Ant Colony Optimization				Results from Derigs et al.		
Dataset	**Objective Function Mean (std dev))**	**Objective Function (Best)**	**Best Time (s)**	**Total Time(s)**	**Objective Function**	**Time(s)**	**GAP**
sd1	240(0)	240	1.743	76.01	**228.28**	300	5.13%
sd2	758(11.35)	740	56.77	87.25	**708.28**	300	4.48%
sd3	451.52(2.42)	447.69	66.12	81.81	**430.58**	300	3.97%
sd4	679.04(1.86)	673.89	65.43	202.75	**631.05**	300	6.79%
sd5	1454.91(3.85)	1445.64	106.92	405.28	**1390.57**	300	3.96%
sd6	860.45(0)	860.45	0.13	378.08	**831.24**	300	3.51%
sd7	3640(0)	**3640**	0.3	603.01	**3640**	300	**0.00%**
sd8	5110.80(45.67)	**5068.28**	214.58	963.57	**5068.28**	300	**0.00%**
sd9	2140.15(14.99)	2129.59	201.15	1017.24	**2067.81**	300	2.99%
sd10	2841.07(14.97)	2807.05	1352.83	2013.42	**2784.21**	300	0.82%
sd11	13280(0)	**13280**	2.65	3086.07	**13280**	300	**0.00%**
sd12	7280.06(0)	7280.06	2337.17	3367.17	**7220.36**	300	0.83%
sd13	10281.74(282.23)	**10171.92**	4653.16	5232.16	10277.81	300	**-1.03%**
sd14	11069.11(46.97)	11021.54	7325.6	9208.81	**10790.58**	3600	2.14%
sd15	15405.92(79.36)	15309.9	12816.82	17594.98	**15152.88**	3600	1.04%
sd16	3411.31(11.17)	3398.69	0.743	17201.99	**3381.29**	3600	0.51%
sd17	26586.11(16.56)	26560.11	12188.12	23866.41	**26536.09**	3600	0.09%
sd18	14772.57(30.52)	14720.11	24301.78	24439.43	**14469.1**	3600	1.73%
sd19	20376.31(29.96)	**20312.44**	11455.71	38677.42	20420.11	3600	**-0.53%**
sd20	40479.27(51.83)	40390.68	49658.4	78854.5	**40368.58**	3600	0.05%
sd21	11449.88(26.31)	11411.61	1.64	121148.8	**11271.06**	3600	1.25%

*****The objective function values highlighted in bold are the best known results
Note: GAP indicates ACO versus previously best known solutions. A negative GAP indicates a new best solution when compared to previous literature.
Derigs et al. (2010) Computer Specifications: 3 GHz, 2 GB Memory, Windows XP.
ACO Computer Specifications: Windows7, Intel i5 2.4 Ghz, 4 GB RAM computer.

The GAP column in Table 2 and Table 3 is the percentage difference in objective function values of ACO and those obtained from Jin et al. (2008) and Derigs et al. (2010) respectively. From Table 2, ACO solutions were between 0.6% - 3.26% of the objective function values from Jin et al. (2008) but the computational times were much faster. Also for 3 datasets, ACO found the best known solutions. For example, in problem s76d2, we found an improved solution that is 1.08% better than the previously best known solution. This problem is a 75 node problem and is one of three problems that the best known solution was improved on in this dataset using the ACO methodology.

However, much greater success was found in improving the best known solutions in the problem sets of Derigs et al. (2010). From Table 3, for 5 out of the 21 datasets, ACO produced better or equivalent results; however this often came at the expense of computational time. For example in problem sd19, ACO was able to find the objective function value 20312.44. This value is 0.53% better than the previously known best solution.

For several of the smaller problems (sd1-sd4), the method appeared to have difficulty. Since these problems consist of fewer than 40 nodes, it was believed that the combination of using a candidate list size of *n/9* and the small problem size may have restricted the algorithm from considering enough nodes in the route construction process. Therefore, in post-hoc testing of these four problems, the candidate list size was removed in order to assess the ability of ACO to solve these smaller problems without the need for a candidate list size. The results of this post-hoc test are listed in Table 4. Notice that after the candidate list was removed, the objective function for sd1 was improved from 240 to 228.28, which is equal to the previously best known solution. Also, as you can see from Table 3 and Table 4, for datasets sd2, sd3 and sd4, a significant improvement in objective function values at the expense of computational time were obtained without using a candidate list. Overall, ACO was able to find improved or equal solutions in 9 out of a total of 32 problem sets.

Table 4: Post-hoc results (without using a candidate list)

		Ant Colony Optimization			Results from Derigs et al.		
Dataset	**Objective Function (Average (std dev))**	**Objective Function (Best)**	**Best Time (s)**	**Total Time(s)**	**Objective Function**	**Time(s)**	**GAP**
sd1	228.28(0)	**228.28**	0.25	27.27	228.28	300	**0.00%**
sd2	747.56(8.86)	734.34	92.53	121.29	**708.28**	300	3.68%
sd3	454.72(6.9)	440.07	48.56	111.11	**430.58**	300	2.20%
sd4	670.18(3.93)	665.94	131.68	270.08	**631.05**	300	5.53%

*****The objective function values highlighted in bold are the best results

As seen from the results in Table 2 and Table 3, ant colony optimization has the ability to find competitive solutions at or within only a few percent of the optimal SDVRP solutions. Also, SDVRP has complex constraints that the memory and learning features of ACO are able to navigate and find improved solutions to, consistent with previous research in the field of logistics on other variants of the VRP. For example, both Bell & Griffis (2010) and Griffis et al. (2012) have shown that the adaptive memory abilities ACO are well-suited to the increasingly complex

constraints of the current VRP instances being analyzed and solved by transportation and supply chain practitioners. However, in our experimental results, for larger problem instance (Table 3), ACO produced better results than the optimal solutions but often at the expense of computational time. Also, the use of candidate lists on larger problems and tuning of ACO parameters significantly improves the ability of ACO to find better solutions.

4.2 Hybrid Metaheuristics Algorithm

Table 5: Hybrid Metaheuristics Parameters

Parameter	Values
Initial Population	500
Size of Future Generation	50
Elite List	5
Number of future generation (Terminating condition)	100

Table 6: Comparing Hybrid metaheuristics algorithm results versus Jin et al.(2008)

	Hybrid Metaheuristics Algorithm			Results from Jin et al.		
Dataset	Objective Function (Average (std dev))	Objective Function (Best)	Time(s)	Objective Function	Total Time(s)	GAP
s51d2	862.67(11.44)	845.86	399.6	**722.93**	10741	17%
s51d3	1118.48(23.45)	1080.32	433.62	**968.85**	833	12%
s51d4	1775.10(15.90)	1752.79	475.56	**1605.64**	789	9%
s51d5	1542.91(14.17)	1512.46	453.6	**1361.24**	10	11%
s51d6	2401.90(1.20)	2398.47	519.12	**2196.35**	478	9%
576d2	1292.75(5.64)	1282.8	756	**1146.68**	75074	12%
s76d3	1674.94(14.12)	1649.51	828	**1474.89**	3546	12%
s76d4	2396.14(24.93)	2357.02	876.6	**2157.87**	369	9%
s101d2	1624.82(20.89)	1586.97	1306.8	**1460.54**	189392	9%
s101d3	2158.10(24.09)	2122.04	1429.2	**1956.91**	36777	8%
s101d5	3134.49(17.22)	3109.88	1539	**2885**	5043	8%

*****The objective function values highlighted in bold are the best results

Table 7: Comparing Hybrid metaheuristics algorithm results versus Derigs et al. (2010)

	Hybrid Metaheuristics Algorithm			Results from Derigs et al.		
Dataset	Objective Function (Average (std dev))	Objective Function (Best)	Time(s)	Objective Function	Time(s)	GAP
sd1	232.38(2.83)	**228.28**	112.56	228.28	300	**0.00%**
sd2	762.83(5.96)	760	165.6	**708.28**	300	7.30%
sd3	466.56(4.86)	458.25	179.1	**430.58**	300	6.43%
sd4	677.05(2.65)	676.28	181.14	**631.05**	300	7.17%
sd5	1520.91(13.68)	1484.85	293.88	**1390.57**	300	6.78%
sd6	860.44(0)	860.44	276.54	**831.24**	300	3.51%
sd7	3640(0)	**3640**	369.24	3640	300	**0.00%**
sd8	5213.19(62.73)	5106.5	492.24	**5068.28**	300	0.75%
sd9	2254.75(25.08)	2206.02	528.36	**2067.81**	300	6.68%
sd10	2853.12(36.29)	**2757.51**	755.28	2784.21	300	**-0.96%**
sd11	13320(28.28)	**13280**	1156.68	13280	300	**0.00%**
sd12	7676.31(31.68)	7627.82	1490.1	**7220.36**	300	5.64%
sd13	10559.42(44.6)	10470.09	1718.52	**10277.81**	300	1.87%
sd14	11399.11(32.14)	11359.9	813.6	**10790.58**	3600	5.28%
sd15	15766.5(56.75)	15681.02	1458	**15152.88**	3600	3.49%
sd16	3397.48(4.34)	3391.7	1090.8	**3381.29**	3600	0.31%
sd17	27532.4(83.43)	27407.36	1863	**26536.09**	3600	3.28%
sd18	15007.04(77.58)	14853.66	1873.62	**14469.1**	3600	2.66%
sd19	20635.12(172.20)	**20260.55**	2972.4	20420.11	3600	**-0.78%**
sd20	41151.15(134.84)	40866.09	5360.88	**40368.58**	3600	1.23%
sd21	11465.5(32.77)	11389.72	28443	**11271.06**	3600	1.05%

*****The objective function values highlighted in bold are the best results

The GAP column in Table 6 and Table 7 is the percentage difference in objective function values of the hybrid metaheuristics algorithm and those obtained from Jin et al. (2008) and Derigs et al. (2010) respectively. The time in Table 6 and Table 7 represents the time stamp at which the best solution was obtained for the proposed hybrid algorithm. From Table 6, the hybrid algorithm was able to find solutions within 8%-17% for all the datasets. However, much greater success was found in improving the best known solutions in the 21 datasets of Derigs et al. (2010). From Table 7, the hybrid algorithm found equal or better solutions for 5 of the 21 datasets (sd1, sd7, sd10, sd11, sd19). For the remaining datasets, the hybrid algorithm found solutions that ranged anywhere between 0.3% to 7.3% of the objective function.

As a further test of the ability of ACO and hybrid metaheuristics algorithm, the ACO objective function values for the two test problem sets are also compared with the dual bound obtained by column generation (Wilck and Cavalier, 2013). These results are shown in Table 8 and Table 9 respectively. The GAP represents the percentage difference between the objective function values of the proposed algorithms and the column generation dual bound. As you can see from Table 8 and Table 9 below, the percentage difference between ACO objective function and column generation dual bound ranges from 0 % to 6.36 % (Derigs et al., 2010) and 3.60% to 8.17% (Jin et al., 2008) respectively whereas for the hybrid metaheuristics algorithm, the percentage difference ranges from 0 % to 7.62 % (Derigs et al., 2010) and 11.81% to 18.56% (Jin et al., 2008) respectively. The dual bound is a computation of the best lower bound for the optimal solution and both algorithms did an excellent job in finding solutions at or close to this bound.

Table 8: Comparison of ACO and hybrid metaheuristics objective function for Derigs et al. (2010) and Column generation dual bound (Wilck and Cavalier, 2013)

Dataset	ACO Objective function	Hybrid Metaheuristics Objective function	Column generation dual bound*	ACO GAP	Hybrid Metaheuristics GAP
sd1	240	**228.28**	228.28	4.88%	0.00%
sd2	740	760	**708.28**	4.29%	6.81%
sd3	447.69	458.25	**430.58**	3.82%	6.04%
sd4	673.89	676.28	**631.05**	6.36%	6.69%
sd5	1445.64	1484.85	**1390.57**	3.81%	6.35%
sd6	860.45	860.44	**831.21**	3.40%	3.40%
sd7	**3640**	**3640**	3640	0.00%	0.00%
sd8	**5068.28**	5106.5	5068.28	0.00%	0.75%
sd9	2129.59	2206.02	**2044.23**	4.01%	7.33%
sd10	2807.05	2757.51	**2684.84**	4.35%	2.64%
sd11	13280	13280	**13265.29**	0.11%	0.11%
sd12	7280.06	7627.82	**7275.97**	0.06%	4.61%
sd13	10171.92	10470.09	**10093.72**	0.77%	3.59%
sd14	11021.54	11359.9	**10632.67**	3.53%	6.40%
sd15	15309.9	15681.02	**15146.92**	1.06%	3.41%
sd16	3398.69	3391.7	**3375.95**	0.67%	0.46%
sd17	26560.11	27407.36	**25320.09**	4.67%	7.62%
sd18	14720.11	14853.66	**14253.94**	3.17%	4.04%
sd19	20312.44	20260.55	**19768.23**	2.68%	2.43%
sd20	40390.68	40866.09	**38071.58**	5.74%	6.84%
sd21	11411.61	11389.72	**11062.32**	3.06%	2.87%

**Column Generation cpu specifications: CPLEX and FORTRAN 95, GNU, Intel Xeon, 2.49 GHz, 8 GB RAM.*
Column Generation stopping criteria: 5% GAP [i.e., GAP = (Primal Solution - Dual Bound) / Primal Solution].

Table 9: Comparison of ACO and hybrid metaheuristics objective function for Jin et al. (2008) and Column generation dual bound (Wilck and Cavalier, 2013)

Dataset	ACO Objective function	Hybrid Metaheuristics Objective function	Column generation dual bound*	ACO GAP	Hybrid Metaheuristics GAP
s51d2	727.28	845.86	**688.83**	5.29%	18.56%
s51d3	982.66	1080.32	**920.58**	6.32%	14.79%
s51d4	1629.09	1752.79	**1520.71**	6.65%	13.24%
s51d5	1389.01	1512.46	**1310.12**	5.68%	13.38%
s51d6	2267.97	2398.47	**2115.2**	6.74%	11.81%
s76d2	1134.27	1282.8	**1093.39**	3.60%	14.77%
s76d3	1502.36	1649.51	**1399.37**	6.86%	15.16%
s76d4	2191.83	2357.02	**2039.11**	6.97%	13.49%
s101d2	1457.39	1586.97	**1395.25**	4.26%	12.08%
s101d3	1948.09	2122.04	**1859.36**	4.55%	12.38%
s101d5	2945.41	3109.88	**2704.63**	8.17%	13.03%

**Column Generation cpu specifications: CPLEX and FORTRAN 95, GNU, Intel Xeon, 2.49 GHz, 8 GB RAM.*
Column Generation stopping criteria: 5% GAP [i.e., GAP = (Primal Solution - Dual Bound) / Primal Solution].

5. Conclusions and Future Directions

In this study, we presented an ACO and a hybrid combination of ACO, GA and a heuristic based approach to solve the Split Delivery Vehicle Routing Problem (SDVRP). The algorithms were tested on benchmark SDVRP test problems and results obtained were promising. For some instances, the best known solution to date was found using the two proposed algorithms. Further, as seen from Table 8 and Table 9, comparing the objective function values for the two proposed algorithms, ACO provided equal or better results for 14 out of 21 datasets in Derigs et al. (2010) and for all datasets in Jin et al. (2008).

An interesting observation that we can highlight and consider for future research is the use of modified candidate list sizes for ACO. As mentioned in previous literature (Bullheimer et al. , 1999a), a candidate list size of one fourth of the total number of customers is recommended, but for our datasets a candidate list of one ninth the total number of customers was found to yield better results during pilot testing. However, at times, this restricted the ability to find improved solutions on the smallest problems. Hence, further research on developing a logic that will generate an ideal candidate list based on total number of customers is needed. Additionally,

practitioners and software developers who use candidate lists in their algorithms should consider how to adjust the candidate list size based on different problem characteristics, and they might consider eliminating the use of a candidate lists in the smallest problems sizes. Also in the future, research should focus on improving the ACO algorithm for SDVRP by (1) using local exchange heuristics to improve the solution, and (2) using specialized groups of ants and multiple colonies as mentioned in the literature (Bell and McMullen, 2004; Gambardella et al., 1999), and considering combining other metaheuristics to solve the SDVRP.

Finally, the successful analysis of the SDVRP in this research using ACO helps confirm the ability of memory based metaheuristics to adapt to the complexities of real-world logistics-oriented vehicle routing problems (Bell and Griffis, 2010). Continued research in the fields of transportation and logistics management should continue to explore how methods such as ACO or a hybrid combination of ACO and other metaheuristics can be used to improve solutions for new instances of the VRP such as the open vehicle routing problem, or it should explore the ability of ACO and hybrid metaheuristics on additional SDVRP constraints such as route congestion, supply chain disruptions of a network link (Griffis et al., 2012) or violations of the triangle inequality caused by traffic delays (Fleming et al., 2012). Doing so will help researchers and practitioners discover the right algorithms and algorithm characteristics needed to solve more complex routing variants such as the SDVRP. Also, in future, we would like to test the two proposed algorithms on other variants of the vehicle routing problem.

References

Aleman, R.E., Zhang, X., Hill, R. R., 2010. An adaptive memory algorithm for the split delivery vehicle routing problem, Journal of Heuristics, 16(3), 441-473.

Archetti, C., Savelsbergh, M., Hertz, A., 2006. A Tabu Search Algorithm for the Split Delivery Vehicle Routing Problem. . Transportation Science, 40(1), 64-73.

Archetti, C., Savelsbergh, M., Speranza, M. G., 2008a. To split or not to split: That is the question. Transportation Research, Part E: Logistics and Transportation Review 44(1), 114-123.

Archetti, C., Speranza, M. G., Savelsbergh, M., 2008b. An Optimization-Based Heuristic for the Split Delivery Vehicle Routing Problem. Transportation Science, 42(1), 22-31.

Archetti, C., Speranza, M. G., 2012. Vehicle routing problems with split deliveries. International Transactions in Operational Research, 19(1-2), 3-22.

Barán, B., Schaerer, M., 2003. A multiobjective ant colony system for vehicle routing problem with time windows. Paper presented at the Proceedings of the 21st IASTED International Conference Applied Informatics, Austria, 97-102.

Bell, J.E., Griffis, S.E., 2010. Swarm intelligence: application of the ant colony optimization algorithm to logistics-oriented vehicle routing problems. Journal of Business Logistics, 31, 157-175.

Bell, J. E., McMullen, P. R., 2004. Ant colony optimization techniques for the vehicle routing problem. Advanced Engineering Informatics, 18, 41-48.

Boudia, M., Prins, C., Reghioui, M., 2007. An effective memetic algorithm with population management for the split delivery vehicle routing problem. Paper presented at the Proceedings of the 4th international conference on Hybrid metaheuristics, Heidelberg

Bullnheimer, B., Hartl, R.F., Strauss, C., 1997. Applying the ant system to the vehicle routing problem. Paper presented at the In Proceedings of the 2nd International Conference on Metaheuristics -MIC97 INRA Sophia-Antipolis & PRiSM, Versailles.

Bullnheimer, B., Hartl, R.F., Strauss, C., 1999. An improved ant system algorithm for the vehicle routing problem. Annals of Operations Research, 89, 319-328.

Chen, S., Golden, B., Wasil, E., 2007. The split delivery vehicle routing problem: Applications, algorithms, test problems, and computational results. Networks, 49(4), 318-327.

Cordeau, J. F., Gendreau, M., Laporte, G., Potvin, J.-Y., & Semet, F., 2002. A guide to vehicle routing heuristics, The Journal of the Operational Research Society, 53(5), 512-522.

Derigs, U., Li, B., Vogel, U., 2010. Local search-based metaheuristics for the split delivery vehicle routing problem. J Oper Res Soc, 61(9),1356–1364.

Doerner, K., Hartl, R. F., Kiechle, G., Lucka, M., Reimann, M., 2004. Parallel ant systems for the capacitated vehicle routing problem. . Paper presented at the Evolutionary Computation in Combinatorial Optimization: 4th European Conference,EvoCOP 2004, Berlin.

Dorigo, M., 1992. Ph.D. Thesis Optimization, learning and natural algorithms (in Italian). Politecnico di Milano,Italy.

Dorigo, M., Gamberdella, L.M., 1997. Ant colonies for traveling salesman problem. BioSystem, 43(1), 73-81.

Dror, M., Trudeau, P., 1989. Savings by split delivery routing. Transportation Science, 23, 141-145.

Dror, M., Laporte, G., Trudeau, P., 1994. Vehicle routing with split deliveries. Discrete Applied Mathematics, 50(3), 239-254.

Dror, M., Trudeau, P., 1990. Split Delivery Routing. Naval Research Logistics 37, 383-402.

Favaretto, D., Moretti, E., Pellegrini, P., 2007. Ant colony system for a VRP with multiple time windows and multiple visits. Journal of Interdisciplinary Mathematics, 10(2), 263-284.

Fleming, C. L., Griffis, S.E., Bell, J.E., 2012. The effects of triangle inequality on the vehicle routing problem. European Journal of Operational Research. In press, accepted manuscript. http://dx.doi.org/10.1016/j.ejor.2012.07.005

Frizzell, P., Giffin, J., 1992. The bounded split delivery vehicle routing problem with grid network distances. Asia-Pacific Journal of Operational Research, 9, 101-116.

Fuellerer, G., Doerner, K. F., Hartl, R. F., Iori, M., 2010. Metaheuristics for vehicle routing problems with three-dimensional loading constraints. European Journal of Operational Research, 201(3), 751-759. doi: 10.1016/j.ejor.2009.03.046

Gajpal, Y., Abad, P., 2009. An ant colony system(ACS) for vehicle routing problem with simultaneous delivery and pickup. Computers & Operations Research, 36, 3215-3223.

Gambardella, L. M., Taillard, E., Agazzi, G., 1999. MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows. In D. Corne, M. Dorigo & F. Glover (Eds.), New Ideas in Optimization (pp. 63-76). UK: McGraw-Hill.

Griffis, S.E., Bell, J.E., Closs, D.J., 2012. Metaheuristics in logistics and supply chain management. Journal of Business Logistics, 32(2), 90-106.

Goldberg, D.E., 1989.Genetic Algorithms in Search, Optimization and Machine Learning. Reading, Addison-Wesley Longman Publishing Co., Inc, Boston, MA.

Hu Meng-Jie, Wang Jian-Ping, Xiao-Min, L., 2011. Solutions to VRP in Home Delivery Based on Ant Colony Optimization Algorithm. Advanced Materials Research, 159, 100-104.

Jin, M., Liua, K., Eksioglu, B., 2008. A column generation approach for the split delivery vehicle routing problem. Operations Research Letters, 36(2), 265-270.

Li, X.Y., Tian, P., Leung, S.C.H., 2009, An ant colony optimizationmetaheuristic hybridized with tabu search for open vehicle routing problems, Journal of the Operational Research Society, 60, 1012-1025.

Liu, S., Huang, W., Ma, H., 2009. An effective genetic algorithm for the fleet size and mix vehicle routing problems. Transportation Research Part E ,45 , 434–445.

Miller, C.E., Tucker, A.W., Zemlin, R.A., 1960. Integer programming formulation of traveling salesman problems, Journal of the ACM, 7(4), 326-329.

Montemanni, R., Gambardella, L. M., Rizzoli, A.E., Donati. A.V., 2004. A new algorithm for a dynamic vehicle routing problem based on ant colony system. Technical Report TR-23-02, IDSIA, Galleria 2. Manno, 6928, Switzerland.

Mota, E., Campos, V., Corberán, Á., 2007. A New Metaheuristic for the Vehicle Routing Problem with Split Demands. Lecture Notes in Computer Science, 4446 121-129.

Rizzoli, A. E., Oliverio, F., Montemanni, R., Gambardella, L. M., 2004. Ant colony optimisation for vehicle routing problem: from theory to applications. Technical Report TR-15-04.

Sui, L.-S., Tang, J.-F., Pan, Z., Liu, S.-A., 2008. Ant colony optimization algorithm to solve split delivery vehicle routing problem. IEEE Control and Decision Conference, Yantai, China, July 2-4, 2008, 997 – 1001.

Wilck, J. H., Rajappa, G., 2010. Ranking Construction Heuristic Solutions for a Hybrid Genetic Algorithm for the Split Delivery Vehicle Routing Problem (SDVRP). Paper presented at the INFORMS Southern Regional Conference 2010, Huntsville, AL.

Wilck, J.H., Cavalier, T.M., 2013. A Column Generation Method for the Split Delivery Vehicle Routing Problem using a Route-Based Formulation, In-Press, International Journal of Operations Research and Information Systems.

Wilck, J.H., Cavalier, T.M., 2012a. A Construction Heuristic for the Split Delivery Vehicle Routing Problem, American Journal of Operations Research, 2(2), 153-162.

Wilck, J.H., Cavalier, T.M., 2012b. A Genetic Algorithm for the Split Delivery Vehicle Routing Problem, American Journal of Operations Research, 2(2), 207-216.

Yi, W., Kumar, A., 2007. Ant colony optimization for disaster relief operations, Transportation Research Part E: Logistics and Transportation Review, 43,660–672.

Yu, B., Yang, Z.Z., 2011. An ant colony optimization model: The period vehicle routing problem with time windows. Transportation Research Part E: Logistics and Transportation Review, 47 (2),166-181.

Yu, B., Yang, Z.Z., Xie, J.X., 2011. A parallel improved ant colony optimization for multi-depot vehicle routing problem. Journal of the Operational Research Society, 62,183-188.